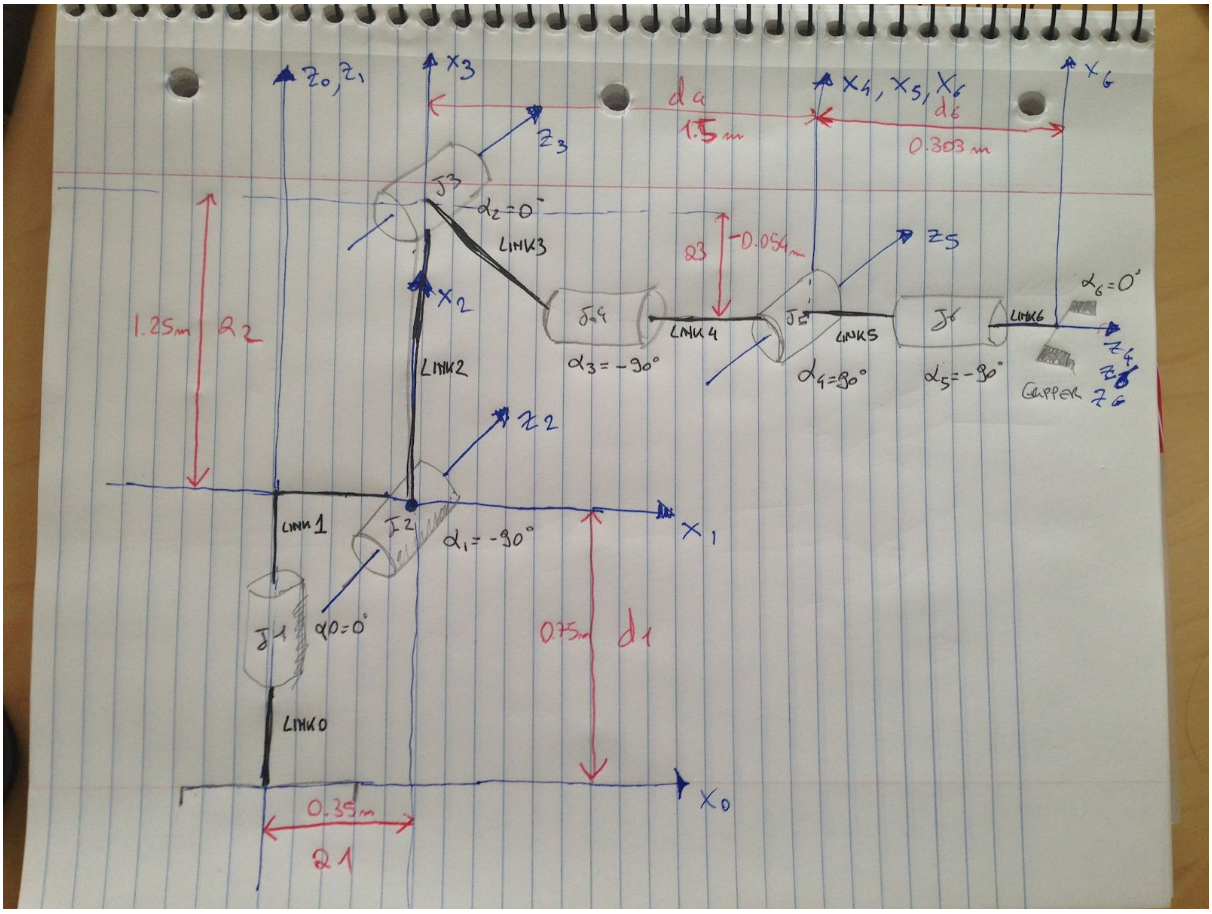
# Kuka KR210 Pick and Place Project

In this project I had the chance to practice with inverse kinematics of a simulated Kuka KR210 arm with 6 degrees of freedom. The arm is meant to pick up an object situated in a random position on a shelf and drop it in a nearby bucket. I have implemented the basic solution as shown in the video walkthrough.

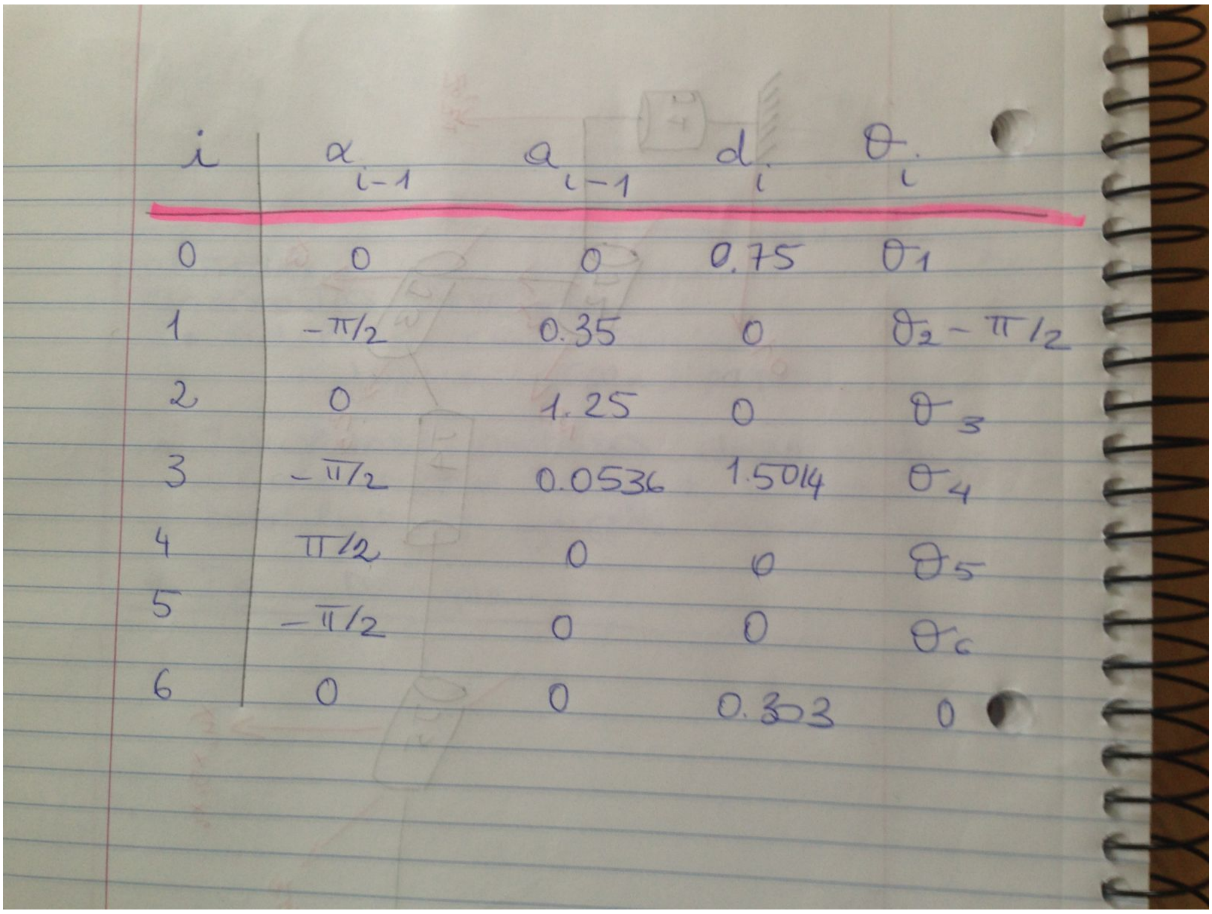
## Kinematic Analysis

### *Denavit-Hartenberg Parameters Table*

Please find below a schematic of the KUKA arm and its DH parameter table. The schematic is drawn in its 0 configuration and was derived on the basis of the algorithm shown in class. In step 1, we label all (revolute) joints from 1 to 6; in step 2, we label links from 0 to  6; in step 3, we draw a line to define all joint axis; in step 4, we define common normal between joint axis; in step 5, we identify the positive verse of z axis along joint axis; in step 6, we identify the positive x axis for intermediate links; in step 7 we define positive x axis for link 0; in step 8, we assign x axis for last link.




The corresponding DH parameter table is shown below



| $i$ | $\alpha_{i-1}$ | $a_{i-1}$ | $d_i$ | $\theta_i$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0.75 | $\theta_1$ |
| 1 | $-\pi/2$ | 0.35 | 0 | $\theta_2 - \pi/2$ |
| 2 | 0 | 1.25 | 0 | $\theta_3$ |
| 3 | $-\pi/2$ | 0.0536 | 1.5014 | $\theta_4$ |
| 4 | $\pi/2$ | 0 | 0 | $\theta_5$ |
| 5 | $-\pi/2$ | 0 | 0 | $\theta_6$ |
| 6 | 0 | 0 | 0.303 | 0 |

Each row in the DH table is a transform from link $i-1$ to $i$ where $\alpha_{i-1}$ is the twist angle between $z_{i-1}$ and $z_i$ measured about $x_{i-1}$ in a right hand sense, $a_{i-1}$ is the distance between $z_{i-1}$ and $z_i$ measured along $x_{i-1}$, link offset is the distance between $x_{i-1}$ and $x_i$ measured along $z_i$ and finally $\theta_i$ (the joint angle) is the angle between $x_{i-1}$ and $x_i$ about the $z_i$ axis. As all joints are revolute, the only non constant value for all transforms is $\theta_i$ and all other parameters can be derived from kr210.urdf.xacro.

*Total Homogeneous Transform between Base Link and Gripper link*

The approach followed to build the complete homogeneous transform from base link to gripper link is the following:

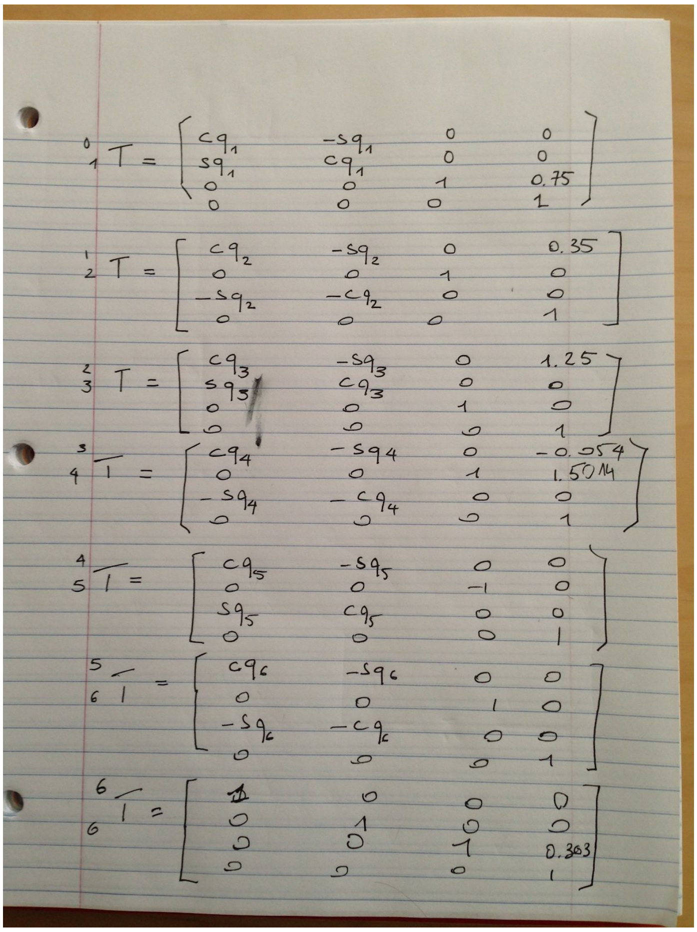
1. define a function that performs a homogeneous transform between neighboring links. This function takes in values from a record in the DH parameter table and is in the form:

$$
{}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

In Python code, the function is:

```
def homogeneous_transform(q, d, a, alpha):
        T = Matrix([[ cos(q),           -sin(q),           0,              a         ],
            [ sin(q)*cos(alpha), cos(q)*cos(alpha), -sin(alpha), -sin(alpha)*d],
            [ sin(q)*sin(alpha), cos(q)*sin(alpha),  cos(alpha),  cos(alpha)*d],
            [                 0,                 0,           0,             1]])
        return T
```

2. compute the homogeneous transform between all neighboring links (between 0 and 1, 1 and 2, and so on):



$$
{}^{0}_{1}T = \begin{bmatrix} cq_1 & -sq_1 & 0 & 0 \\ sq_1 & cq_1 & 0 & 0 \\ 0 & 0 & 1 & 0.75 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{1}_{2}T = \begin{bmatrix} cq_2 & -sq_2 & 0 & 0.35 \\ 0 & 0 & 1 & 0 \\ -sq_2 & -cq_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{2}_{3}T = \begin{bmatrix} cq_3 & -sq_3 & 0 & 1.25 \\ sq_3 & cq_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{3}_{4}T = \begin{bmatrix} cq_4 & -sq_4 & 0 & -0.054 \\ 0 & 0 & 1 & 1.5014 \\ -sq_4 & -cq_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{4}_{5}T = \begin{bmatrix} cq_5 & -sq_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sq_5 & cq_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{5}_{6}T = \begin{bmatrix} cq_6 & -sq_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sq_6 & -cq_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{6}_{G}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.303 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

3. compute the generalized homogeneous transform from link 0 to gripper link by multiplying the homogeneous transforms in step 2
   $T0\_G \;=\; T0\_1 * T1\_2 * T2\_3 * T3\_4 * T4\_5 * T5\_6 * T6\_G$

## *Inverse Kinematics Problem*

1. *Calculate the location of the wrist centre* ( $O4 = O5 = O6$ )

As seen in class, the wrist-centre coordinates with respect to the base frame are given by:

$$w_x = p_x - (d_6 + l) \cdot n_x$$
$$w_y = p_y - (d_6 + l) \cdot n_y$$
$$w_z = p_z - (d_6 + l) \cdot n_z$$

where $px, py, pz$ are given by the simulator, $d6$ is given by the DH parameter table and $l$ is the end effector length. $nx, ny, nz$ are the values of the $n$ vector representing the end effector orientation along the $z$ axis of the local coordinate frame. We can obtain the components of vector $n$ from the third column of the rotation matrix of the end effector with respect to the base frame as follows

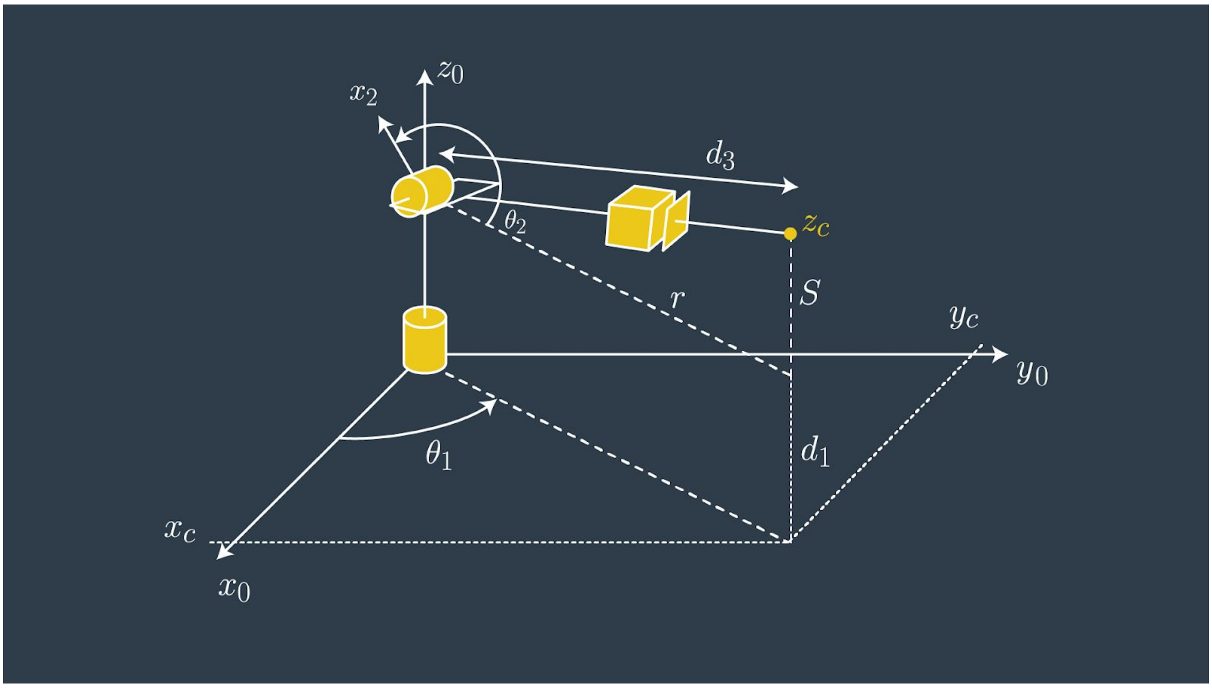
```
Rrpy = Rot(Z, yaw) * Rot(Y, pitch) * Rot(X, roll) * R_corr
```

Here, yaw, pitch and roll are given by the simulator and R_corr is the correctional rotation matrix calculated to account for the difference in orientation between the URDF file and the DH convention.

2. *Use trigonometry to solve* $\theta_1$ *to* $\theta_3$

$\theta_1$ is calculated by projecting the wrist centre into the plane $X_0Y_0$ (as seen in the class), thus it is

$$\theta_1 = atan2(y_c, x_c)$$



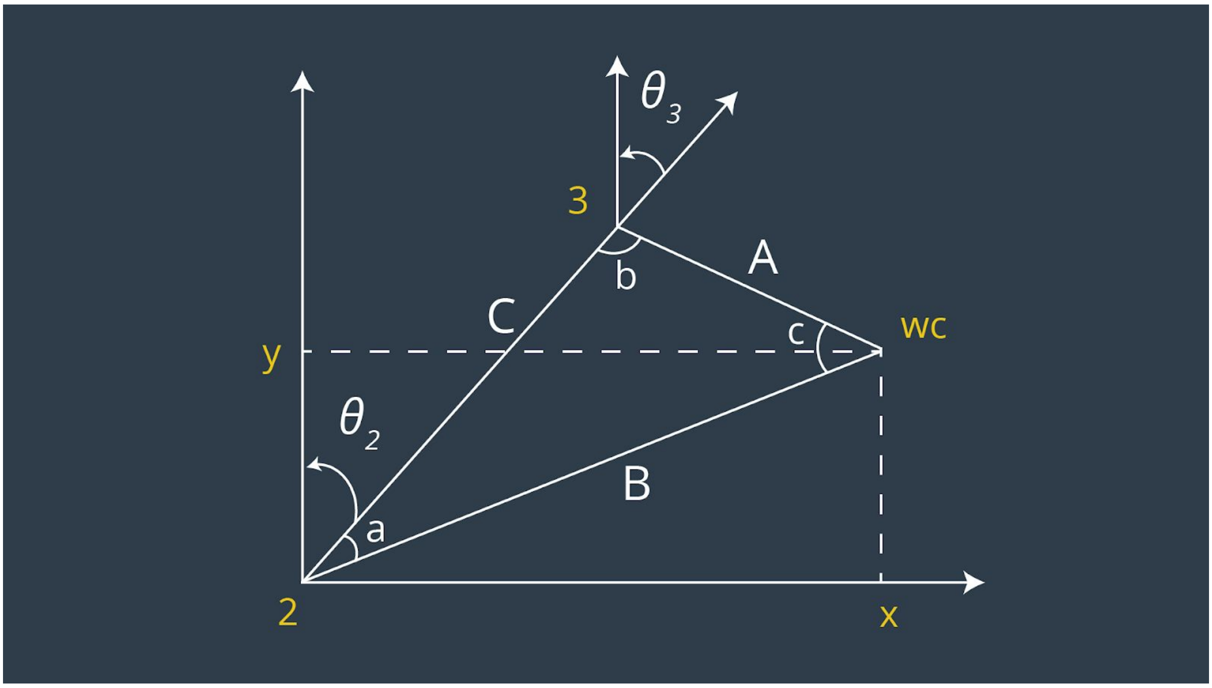

$\theta_2$ and $\theta_3$ can be calculated as follows:

$\theta_2 = \pi/2 - a - atan2(WC_z - d_1, \sqrt{}(WCx^2 + WCy^2) - a_1)$

where $a$ can be deduced from the law of cosines, once we have calculated B (A and C are from DH parameter table)

$B = \sqrt{}((sqrt(WCx^2 + WCy^2)^2 - a_1) + (WCz - d_1)^2)$

$\theta_3 = \pi/2 - (b + 0.036)$



3. *Calculate $\theta_4$ to $\theta_6$ using Euler angles from rotation matrix*

To calculate the last 3 joint variables, we need to set Rrpy rotation between base_link and gripper_link calculated above equal to R0_6 (=R0_1* R1_2 *...R5_6)

```
R0_1*R1_2*R2_3*R3_4*R4_5*R5_6 = Rrpy
```

pre-multiply both sides of the above equation by **inv(R0_3)**

```
R3_6 = inv(R0_3) * Rrpy
```

`R0_3` can be calculated as

```
R0_3 = T0_1[0:3,0:3] * T1_2[0:3,0:3] * T2_3[0:3,0:3]
```

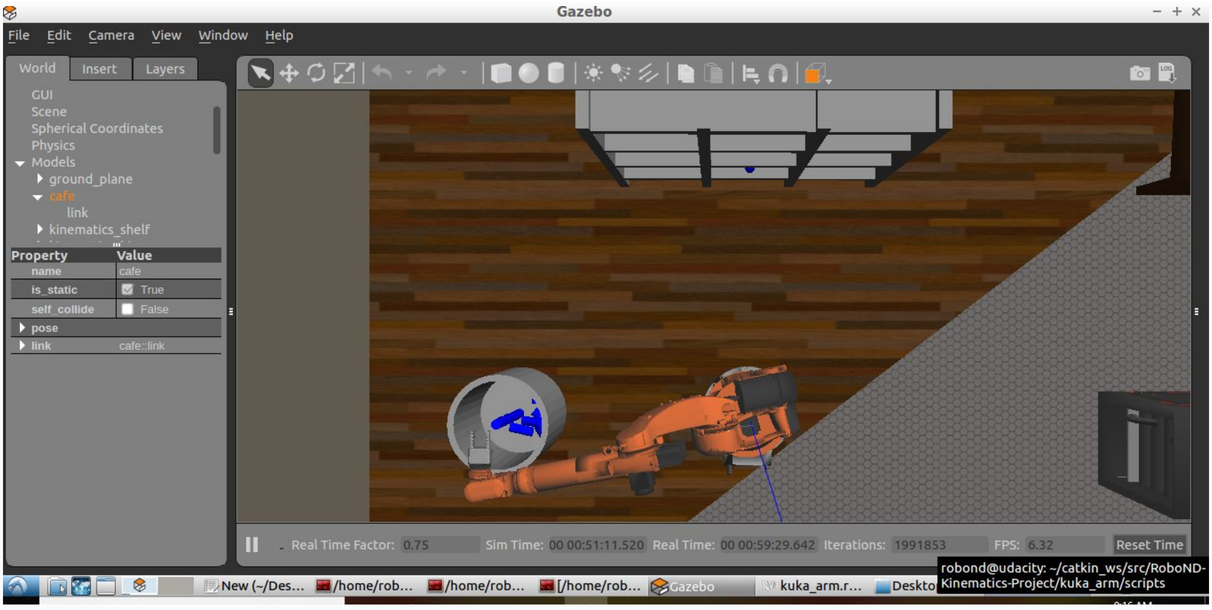
and by substituting the newly calculated joint angles 1-3.

$\theta_4$ *to* $\theta_6$ were calculated as Euler angles from a rotation matrix.

# Results

The manipulator was successful in retrieving the target object from the shelf and dropping it in the dropping area 9 out of 10 times. In one occasion, it moved empty handed and did not grasp the target. The calculated trajectory sometimes contained loops that were not necessary, resulting in unnecessary rotations of the manipulator and a slowing down of the process.

Please find below a view from the top of the scene at the end of the 10th run and a close up of the bin.


Gazebo
File
Edit
Camera
View
Window
Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
ground_plane
cafe
link
kinematics_shelf
Property
Value
name
cafe
is_static
True
self_collide
False
pose
link
cafe::link
Real Time Factor:
0.75
Sim Time:
00 00:51:11.520
Real Time:
00 00:59:29.642
Iterations:
1991853
FPS:
6.32
Reset Time
New (~/Des...
/home/rob...
/home/rob...
[/home/rob...
Gazebo
kuka_arm.r...
Deskto
robond@udacity: ~/catkin_ws/src/RoboND-Kinematics-Project/kuka_arm/scripts